Problem 1

Show that there exists a Turing machine that on input 0^i1^j0 outputs 0^i 1^j 0 for all i, j ∈ N. Describe how the Turing machine operates on the tape for a given input 0^i 1^j 0.

In all following problems, you may assume that there similarly exist Turing machines that compute other basic arithmetic and control-flow operations, such as addition, if-then-else, and while loops. You may therefore provide pseudocode whenever a description of a Turing machine is requested.

Problem 2

a. Show that there exists a Turing machine to decide if the language of a deterministic finite automaton is empty. In other words, show that the following language is decidable,

\[ L_1 = \{ (A) \mid A \text{ is a DFA with } L(A) = \emptyset \} \]

b. Show that there exists a Turing machine to decide given two DFAs A and B if the language of A contains the language of B. In other words, show that the following language is decidable,

\[ L_2 = \{ (A, B) \mid A \text{ and } B \text{ are DFAs with } L(A) \subseteq L(B) \} \]

*Hint:* Use closure properties of regular languages to reduce \( L_2 \) to \( L_1 \).
Problem 3

Show that the following languages are undecidable. You may use Rice’s theorem if it applies or you may reduce from the acceptance problem $A_{TM}$ for Turing machines.

a. $L_1 = \{(M, w) \mid M$ is a single-tape TM that never modifies the portion of the tape that contains the input $w\}$,

b. $L_2 = \{(M) \mid M$ is a TM and $L(M) = \Sigma^*\}$.

Problem 4

We say a function $f : \mathbb{N} \to \mathbb{N}$ is **computable** if there exists a Turing machine $M$ that on input $1^n$ outputs\(^1\) $1^{f(n)}$. (You may assume that $M$ has tape alphabet $\{0, 1, \square\}$.)

A **busy beaver** with $n$ states is a Turing machine with tape alphabet $\{0, 1, \square\}$, which on input $\varepsilon$ outputs\(^1\) a string of as many 1s as possible (among all such Turing machines with $n$ states). For $n \in \mathbb{N}$, let $BB(n)$ be the number of 1s a busy beaver with $n$ states can produce.

a. Prove that BB is nondecreasing, i.e. $BB(m) \geq BB(n)$ if $m \geq n$.

b. Argue that for every computable function $f$, there exists a constant $c \in \mathbb{N}$ such that $\forall n \in \mathbb{N}. BB(c + n) \geq f(BB(n))$.

c. Show that $BB(n) \geq 2 \cdot n - d$ for some constant $d \in \mathbb{N}$.

d. Hence conclude that $BB(n)$ is not computable.

Remark: Each of the four parts has a short answer.

---

\(^1\)Recall that we say that a Turing machine $M$ on input $w$ **outputs** a string $w'$ if $M$ halts with just $w'$ written on the tape.